

BUL in MediaEval 2016 Emotional Impact of Movies Task

Asim Jan, Yona Falinie A. Gaus, Fan Zhang, Hongying Meng
Department of Electronic and Computer Engineering, Brunel University, London
{asim.jan, yonafalinie.abdgaus, fan.zhang, hongying.meng}@brunel.ac.uk

ABSTRACT

This paper describes our working approach for the Emotional Impact of Movies task of MediaEval 2016. There are 2 sub-tasks set to make affective predictions, based on Arousal and Valence values, on video clips. Sub-task 1 requires global emotion prediction. Here a framework is developed using Deep Auto-Encoders, a feature variation algorithm and a Deep network. For sub-task 2, a set of audio features are extracted for continuous emotion prediction. Both sub-tasks are approached as a regression problem evaluated by Mean Squared Error and Pearson Correlation Coefficient.

1. INTRODUCTION

The 'Emotional Impact of Movies Task' comprises two sub-tasks with the goal of creating a system that automatically predicts the emotional impact on video contents, in terms of Arousal and Valence, which in a 2-D scale can be used to describe emotions. **Sub-task 1** - Global emotion prediction, predicting a score on induced Valence (negative-positive) and induced Arousal (calm-excited) for the whole clip; **Sub-task 2** - Continuous emotion prediction, predicting a score of induced Arousal and Valence continuously for each 1s-segment of the video. The development dataset used in both task is the LIRIS-ACCEDE dataset [2]. For the first sub-task, 9800 video excerpts (around 10s) are provided with the global Valence and Arousal annotations. For the second sub-task, 30 movies are provided with the continuous annotation of Valence and Arousal. Full details on the challenge tasks and database can be found in [3].

2. METHODOLOGY

2.1 Framework Summary for Sub-task 1:

The framework is primarily based on visual cues, with the use of Deep Learning to benefit from the large sample video dataset. The content of the videos have many different scenes making the emotion detection process challenging. To tackle this, a 3 stage framework has been designed. The first stage of the framework is a Deep Auto-Encoder, which can be understood in [1]. This is utilized to try and give a representation recreated by a Deep Network. It is trained with all video samples and each image is reproduced

at frame level to try and obtain common representations amongst all the videos. It is likely that this will highlight peoples faces and hide the uncommon scenes and objects. The second stage observes the decoded features for variations within a video sample by using Feature Dynamic History Histogram (FDHH) across the frame level, to produce a histogram of patterns that summarize and capture these observations from a set of features. Finally the FDHH features are used with a regressive model to predict the Arousal and Valence scales.

2.1.1 Stage 1 - Auto-Encoder:

There are two Deep Auto-Encoders trained, one based on MSE loss and the other on Euclidean loss. Using a similar architecture of Fig. 1 found in [10], the Auto-Encoders both have the same architecture of 4 convolution (Conv) layers followed by 4 deconvolution (DeConv) layers. Each of the Conv and DeConv layers are followed by a Rectified Linear Unit (ReLU) activation layer, and at the end is a loss layer.

2.1.2 Stage 2 - FDHH Feature:

The FDHH algorithm, based on the idea of Motion History Histogram (MHH) [8], aims to extract temporal movement across the feature space. This is achieved firstly by taking the absolute difference of a feature vector $V(n, 1 : c)$ representing a frame, and its following frame $V(n+1, 1 : c)$ to produce $D(n-1, 1 : c)$, where n is the frame sample and c is the feature dimension. Next, the result calculated of each dimension from the vector D is compared to a threshold T that is set by the user to control the amount of variation to detect, producing a vector of 1's and 0's, that represent above and below the threshold. This is repeated for all frames except the last frame, and a new feature set $F(1 : N-1, 1 : C)$ is produced. Next, each dimension c is observed for patterns $m = 1 : M$ throughout the feature vector $F(1 : N-1, c)$, where a histogram is then produced for each defining pattern. A pattern can be defined as the number of consecutive 1's e.g. $m = 1$ would look for a pattern "010", and $m = 2$ would look for '0110'. The final FDHH Feature will of dimensions $FDHH(1 : M, 1 : C)$.

2.1.3 Stage 3 - Regression Models:

The final stage of the framework is the regression model, in which 2 are utilized. First being a Deep Network trained on the FDHH features using MSE and Euclidean Regression loss function, and the other treats this trained Deep Network as a Pre-Trained feature extractor, and applies Partial Least Squares (PLS) on the features to predict the Arousal and Valence values.

The Deep Network consists of 9 Conv layers, 1 Pooling layer, 8 ReLu Activation Layers and a Loss Layer at the end. Training is done using Support Gradient Descent until 100 epochs, with the weights initialized using the Xavier method [5].

The Pre-Trained features are extracted from the 100th epoch network, which are concatenated with Audio descriptors that are mentioned in Section 2.2.1, however they are based on a whole video clip samples rather than 1s segments. These features are concatenated, Rank Normalized between 0 and 1, and then PLS regression is used.

2.2 Framework Summary for Sub-task 2:

2.2.1 Stage 1 - Audio Descriptors:

The Audio descriptors are extracted from openSMILE software [9] which include 16 low-level descriptor (LLD) as follows: root mean square (RMS) frame energy; zero-crossing-rate (ZCR); harmonics-to-noise ratio (HNR); pitch frequency (F0); mel-frequency cepstral coefficients (MFCC) 1-12. For each LLDs, 12 functionals mean, standard deviation, kurtosis, skewness, minimum and maximum value, relative position, and range as well as two linear regression coefficients with their mean square error (MSE) are also computed. In total, the number of features per 1s-segment are $(16 \times 2 \times 12) = 384$ attributes.

2.2.2 Stage 2 - Regression Models:

A total of 3 regressive models are trained on the audio descriptors. These are mentioned in the following:

Run 1 Linear Regression + Gaussian smoothing (LR+Gs): After obtaining the predicted label from the regression stage, a smoothing operation is performed, using Gaussian filtering with a window size of 10. The smoothing window is carefully selected, in order to retain the pattern of the labels whilst increasing the performance. It is required for removing the high frequency noise irrelevant to the affective dimensions.

Run 2 Partial Least Square (PLS): PLS is a statistical algorithm that bears some relation to principal components regression. Previous EmotiW 2015 employed PLS in the systems, which gave better results than the baseline [6] [7].

Run 3 Least Square Boosting + Moving Average smoothing (LSB + MAs): LSB is regression model trained with gradient boosting [4]. In this model, the number of regression trees in the ensemble is chosen as 500 on a training set. After obtaining the prediction labels, a smoothing operation is performed using a moving average filter, in order to increase the performance.

3. EXPERIMENTAL SETUP

5 different runs were made based on the framework Sub-task 1, and 3 different runs for Sub-task 2, which are:

Run 1 & Run 2 Deep_Audio_PLS: These runs utilize a trained Auto-Encoder with a Euclidean loss function (EUC_Loss) and MSE loss function (MSE_Loss), followed by FDHH feature extraction. Finally trained Deep Networks, with Euclidean and MSE loss respectively, are used as a Pre-Trained Feature Extractors. These features are fused with Audio features and a PLS regression model is trained.

Run 3 Audio + PLS: This run is based on using just the openSMILE Audio descriptors, along with a PLS regression

Table 1: Results on sub-task 1 using the proposed framework

Run	Arousal		Valence	
	MSE	PCC	MSE	PCC
1	1.462	0.251	0.236	0.143
2	1.441	0.271	0.237	0.144
3	1.525	0.143	0.236	0.125
4	1.443	0.248	0.231	0.146
5	1.431	0.263	0.231	0.149

Table 2: Results on sub-task 2 using proposed framework

Run	Arousal		Valence	
	MSE	PCC	MSE	PCC
1	0.157	-0.072	0.173	-0.042
2	0.129	-0.013	0.141	-0.007
3	0.182	-0.039	0.175	-0.062

model.

Run 4 and Run 5 Deep Network: This run is based on training an Auto-Encoder with a Euclidean loss function (EUC_Loss) and MSE loss function (MSE_Loss). They are also followed by FDHH feature extraction, with Deep Networks trained on the FDHH features as a regressive model, using Euclidean and MSE regressive loss functions respectively.

For Sub-Task 2, for each configuration, 3 different runs were selected, as explained in Section 2.2.

4. RESULTS AND DISCUSSIONS

On the official test results, each of the sub-task were evaluated using Mean Squared Error (MSE) and Pearson Correlation Coefficient (PCC). For Sub-Task 1, Table 1, the results show a strong performance for Run 5, using a trained Deep Network with Euclidean loss and no Audio descriptors. It is closely matched by Run 4, the identical configuration using MSE loss to train the Deep Networks. The Audio descriptors have shown the weakest performance of them all, with the possibility of increasing the errors of Runs 1 and 2, as they use Audio fusion. In terms of Training loss functions (EUC VS MSE), when comparing Runs 1 VS 2 and Runs 4 VS 5, there is a performance boost for Euclidean loss in most cases, but only marginal. For Sub-Task 2, Table 2, PLS gives lowest MSE but LR+Gs gives highest results on PCC. However all algorithms perform unacceptably bad on Valence, a situation that requires further investigation.

5. CONCLUSIONS

In this working notes paper, we proposed a different framework for each sub-task. The frameworks are composed on feature extraction using deep learning, FDHH for capturing Feature variations across the Deep Features at frame level, and finally audio descriptors taken from the speech signal. Several machine learning algorithms were also implemented as a regression model. The official test results show that features proposed by the framework are informative, give good results in terms of MSE and PCC in sub-task 1 and good results in terms of MSE in sub-task 2. The future work will focus on the dynamic relationship of the emotion data.

6. REFERENCES

- [1] P. Baldi. Autoencoders, unsupervised learning, and deep architectures. In *Unsupervised and Transfer Learning - Workshop held at ICML 2011, Bellevue, Washington, USA, July 2, 2011*, pages 37–50, 2012.
- [2] Y. Baveye, E. Dellandréa, C. Chamaret, and L. Chen. LIRIS-ACCEDE: A video database for affective content analysis. *IEEE Transactions on Affective Computing*, 6(1):43–55, 2015.
- [3] E. Dellandréa, L. Chen, Y. Baveye, M. Sjöberg, C. Chamaret, and E. C. D. Lyon. The MediaEval 2016 Emotional Impact of Movies Task. pages 3–5, 2016.
- [4] J. H. Friedman. Stochastic gradient boosting. *Computational Statistics and Data Analysis*, 38(4):367–378, 2002.
- [5] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS10). Society for Artificial Intelligence and Statistics*, 2010.
- [6] H. Kaya and A. A. Salah. Contrasting and Combining Least Squares Based Learners for Emotion Recognition in the Wild Contrasting and Combining Least Squares Based Learners for Emotion Recognition in the Wild. (November):459–466, 2015.
- [7] M. Liu, R. Wang, Z. Huang, S. Shan, and X. Chen. Partial least squares regression on grassmannian manifold for emotion recognition. *Proceedings of the 15th ACM on International conference on multimodal interaction - ICMI '13*, pages 525–530, 2013.
- [8] H. Meng, N. Pears, M. Freeman, and C. Bailey. Motion history histograms for human action recognition. In *Embedded Computer Vision*, pages 139–162. 2009.
- [9] F. Weninger. open-Source Media Interpretation by Large feature-space Extraction. (December), 2014.
- [10] Y. Zhou, D. Arpit, I. Nwogu, and V. Govindaraju. Is Joint Training Better for Deep Auto-Encoders? *ArXiv e-prints*, May 2014.